**Aula 1 – Introdução a Sistemas Operacionais**

**Sistema Operacional**

Software cuja função é fornecer aos programas do usuário um modelo do computador melhor, mais simples e mais limpo, assim como lidar com o gerenciamento do hardware. Exemplos: Windows, Linux, Unix, FreeBSD, Solaris, entre outros.

**O sistema operacional como uma máquina estendida**

Os clientes reais dos sistemas operacionais são os programas aplicativos. São eles que lidam diretamente com as abstrações fornecidas pela interface do usuário, seja uma linha de comandos (shell) ou uma interface gráfica.

**O sistema operacional como um gerenciador de recursos**

Fornece uma alocação ordenada e controlada dos processadores, memórias e dispositivos de E/S entre os vários programas competindo por eles.

**O gerenciamento de recursos inclui a multiplexação (compartilhamento) de recursos de duas maneiras diferentes: no tempo e no espaço.**

Quando um recurso é multiplexado no tempo, diferentes programas ou usuários se revezam usando-o. Primeiro, um deles usa o recurso, então outro e assim por diante.

O outro tipo é a multiplexação de espaço. Em vez de os clientes se revezarem, cada um tem direito a uma parte do recurso. Por exemplo, a memória principal é normalmente dividida entre vários programas sendo executados, de modo que cada um pode ser residente ao mesmo tempo (por exemplo, a fim de se revezar usando a CPU).

**Multiprogramação.**

Na multiprogramação, a memória era dividida em várias partes, com uma tarefa diferente em cada partição. Enquanto uma tarefa ficava esperando pelo término da E/S, outra podia usar a CPU. Se um número suficiente de tarefas pudesse ser armazenado na memória principal ao mesmo tempo, a CPU podia se manter ocupada quase 100% do tempo

**Sistemas de tempo compartilhado (time-sharing systems)**

Nos sistemas de tempo compartilhado, o controle da execução dos programas é feito interativamente e ocorre a eliminação do monopólio sobre a CPU da multiprogramação, realizando uma distribuição mais justa do tempo de uso do processador.

**Revisão sobre hardware de computadores**

* **Processadores**

O “cérebro” do computador é a CPU. Ela busca instruções da memória e as executa.

* **Memória principal (Memória RAM)**

O segundo principal componente em qualquer computador é a memória. Uma memória deve ser rápida ao extremo (mais rápida do que executar uma instrução, de maneira que a CPU não seja atrasada pela memória), abundantemente grande e muito barata.

* **Discos**

Componente responsável pelo armazenamento permanente dos dados.

* **Dispositivos de E/S (Entrada e Saída)**

Permitem a entrada de dados e sua exibição (informação) após o processamento. Exemplo: Mouse, teclado, microfone (entrada). Monitor, caixa de som (saída).

* **Barramentos**

Auxiliam no tráfego de dados entre os dispositivos de hardware na parentboard (placa-mãe). Exemplos: USB, SATA, PCIe (PCI Express).

**Aula 2 - Tipos e conceitos de Sistemas Operacionais**

**Tipos de Sistemas Operacionais**

**Sistemas operacionais de computadores de grande porte**

São intensamente orientados para o processamento de muitas tarefas ao mesmo tempo, a maioria delas exigindo quantidades extraordinárias de E/S.

**Sistemas operacionais de servidores**

São executados em servidores que são computadores pessoais muito grandes, em estações de trabalho ou mesmo computadores de grande porte. Servem a múltiplos usuários ao mesmo tempo por meio de uma rede e permitem que os usuários compartilhem recursos de hardware e software.

**Sistemas operacionais de multiprocessadores**

Sistemas operacionais com aspectos especiais para comunicação, conectividade e consistência para máquinas com vários processadores.

**Sistemas operacionais de computadores pessoais**

São executados em servidores que são computadores pessoais muito grandes, em estações de trabalho ou mesmo computadores de grande porte. Servem a múltiplos usuários ao mesmo tempo por meio de uma rede e permitem que os usuários compartilhem recursos de hardware e software.

**Sistemas operacionais embarcados**

São executados em computadores que controlam dispositivos que não costumam ser vistos como computadores e que não aceitam softwares instalados pelo usuário.

**Sistemas operacionais de nós sensores (sensor-node)**

São executados em computadores pequenos movidos a bateria com rádios integrados. Comunicam entre si e com uma estação-base usando comunicação sem fio

**Sistemas operacionais de tempo real**

São caracterizados por ter o tempo como um parâmetro chave. Muitos desses sistemas são encontrados no controle de processos industriais, aviônica, militar e áreas de aplicação semelhantes. Esses sistemas têm de fornecer garantias absolutas de que uma determinada ação ocorrerá em um determinado momento.

**Sistemas operacionais de cartões inteligentes (smartcard)**

Os menores sistemas operacionais são executados em cartões inteligentes, que são dispositivos do tamanho de cartões de crédito contendo um chip de CPU. Possuem severas restrições de memória e processamento de energia.

**Conceitos**

**Processos**

Conceito fundamental em todos os sistemas operacionais é o processo. Um processo é basicamente um programa em execução.

**Espaços de endereçamento**

Conjunto de endereços na memória principal que podem ser usados por um processo gerenciados pelo Sistema Operacional

**Arquivos**

Forma abstrata de representação de um conjunto de dados ou instruções.

**Aula 3 - Estrutura do Sistema Operacional**

**Núcleo do sistema, ou kernel.**

Conjunto de rotinas que oferece serviços aos usuários e às suas aplicações.

**Principais funções do núcleo**

Tratamento de interrupções e exceções; Criação e eliminação de processos e threads; Sincronização e comunicação entre processos e threads; Escalonamento e controle dos processos e threads; Gerência de memória; Gerência do sistema de arquivos; Gerência de dispositivos de E/S; Suporte a redes locais e distribuídas; Contabilização do uso do sistema; Auditoria e segurança do sistema.

**Modo de Acesso**

As instruções privilegiadas só podem ser executadas quando o modo de acesso do processador encontra-se em kernel, caso contrário o hardware irá impedir a execução da instrução.

As instruções não privilegiadas são as que não oferecem risco ao sistema e podem ser executadas em modo não privilegiado, ou seja, modo usuário.

**Rotinas do Sistema**

As rotinas do sistema operacional compõem o núcleo do sistema, oferecendo serviços aos usuários e suas aplicações.

Todas as funções do núcleo são implementadas por rotinas do sistema que necessariamente possuem em seu código instruções privilegiadas. A partir desta condição, para que estas rotinas possam ser executadas o processador deve estar obrigatoriamente em modo kernel, o que exige a implementação de mecanismos de proteção para garantir a confiabilidade do sistema.

Todo o controle de execução de rotinas do sistema operacional é realizado pelo mecanismo conhecido como system call.

**System Call**

Mecanismo de proteção por software, no qual o sistema operacional garante que as aplicações só poderão executar rotinas do sistema que estão previamente autorizadas.

**Arquiteturas do Núcleo**

**Arquitetura Monolítica**

Aplicação formada por vários módulos, formando um grande e único programa executável, onde os módulos podem interagir livremente.

**Arquitetura de Camadas**

Aplicação onde o sistema é dividido em níveis sobrepostos. Cada camada oferece um conjunto de funções que podem ser utilizadas apenas pelas camadas superiores. Neste tipo de implementação, as camadas mais internas são mais privilegiadas que as mais externas.

**Máquina Virtual**

O modelo de máquina virtual, ou virtual machine (VM), cria um nível intermediário entre o hardware e o sistema operacional, denominado gerência de máquinas virtuais. Este nível cria diversas máquinas virtuais independentes, onde cada uma oferece uma cópia virtual do hardware, incluindo os modos de acesso, interrupções, dispositivos de E/S, etc.

**Arquitetura Microkernel**

A arquitetura microkernel permite isolar as funções do sistema operacional por diversos processos servidores pequenos e dedicados a serviços específicos, tornando o núcleo menor, mais fácil de depurar e, consequentemente, aumentando sua confiabilidade. Na arquitetura microkernel, o sistema operacional passa a ser de mais fácil manutenção, flexível e de maior portabilidade.

**Aula 4 - Processo**

**Definição de processo de forma mais detalhada**

Para que a concorrência entre os programas ocorra sem problemas, é necessário que todas as informações do programa interrompido sejam guardadas para que, quando este voltar a ser executado, não lhe falte nenhuma informação necessária à continuação do processamento. Estas informações são fundamentais para que o sistema operacional possa gerenciar a execução concorrente de programas, e é a base de qualquer ambiente multiprogramável. O conceito de processo pode ser definido como sendo o conjunto necessário de informações para que o sistema operacional implemente a concorrência de programas.

**Estrutura do Processo**

**Contexto de Hardware**

Armazena o conteúdo dos registradores gerais da UCP, além dos registradores de uso específico.

**Contexto de Software**

São especificados limites e características dos recursos que podem ser alocados pelo processo.

**Espaço de Endereçamento**

O espaço de endereçamento é a área de memória pertencente ao processo onde instruções e dados do programa são armazenados para execução.

**Estados do Processo**

Em um sistema multiprogramável, um processo não deve alocar exclusivamente a UCP, de forma que exista um compartilhamento no uso do processador. Os processos passam por diferentes estados ao longo do seu processamento, em função de eventos gerados pelo sistema operacional ou pelo próprio processo. Um processo ativo pode encontrar-se em três diferentes estados: Execução, Pronto e Espera.

Execução (running) : um processo é dito no estado de execução quando está sendo processado pela UCP.

Pronto (ready) : Um processo está no estado de pronto quando aguarda apenas para ser executado.

Espera (wait): Um processo no estado de espera aguarda por algum evento externo ou por algum recurso para prosseguir seu processamento.

**Processos CPU-bound e I/O-bound**

Processos podem ser classificados como CPU-bound ou I/O-bound de acordo com a utilização do processador e dos dispositivos de E/S.

Um processo é definido como CPU-bound (ligado à UCP) quando passa a maior parte do tempo no estado de execução, utilizando o processador, ou pronto. Esse tipo de processo realiza poucas operações de leitura e gravação, e é encontrado em aplicações científicas que efetuam muitos cálculos.

Um processo é classificado como I/O-bound (ligado à E/S) quando passa a maior parte do tempo no estado de espera, pois realiza um elevado número de operações de E/S. Esse tipo de processo é encontrado em aplicações comerciais, que se baseiam em leitura, processamento e gravação.

**Processos Foreground e Background**

Um processo foreground é aquele que permite a comunicação direta do usuário com o processo durante o seu processamento. Neste caso, tanto o canal de entrada quanto o de saída estão associados a um terminal com teclado, mouse e monitor, permitindo, assim, a interação com o usuário.

Um processo background é aquele onde não existe a comunicação com o usuário durante o seu processamento). Neste caso, os canais de E/S não estão associados a nenhum dispositivo de E/S interativo, mas em geral a arquivos de E/S.

**Aula 5 – Thread**

**Thread**

Um thread pode ser definido como uma subrotina de um programa que pode ser executada de forma assíncrona, ou seja, executada concorrentemente ao programa chamador.

**Aplicação monothread**

Em um ambiente monothread, um processo suporta apenas um programa no seu espaço de endereçamento. Aplicações concorrentes são implementadas apenas com o uso de múltiplos processos independentes ou subprocessos.

**Aplicação multithread**

No ambiente multithread, cada processo pode responder a várias solicitações concorrentemente ou mesmo simultaneamente, caso haja mais de um processador. A grande vantagem no uso de threads é a possibilidade de minimizar a alocação de recursos do sistema, além de diminuir o overhead (sobrecarga) na criação, troca e eliminação de processos.

**Aula 6 – Gerenciamento de memória**

**Objetivo do gerenciamento de memória pelo Sistema Operacional**

Enquanto nos sistemas monoprogramáveis a gerência da memória não é muito complexa, nos sistemas multiprogramáveis ela se torna crítica, devido à necessidade de se maximizar o número de usuários e aplicações utilizando eficientemente o espaço da memória principal.

**Swapping**

A gerência de memória deve tentar manter na memória principal o maior número de processos residentes, permitindo maximizar o compartilhamento do processador e demais recursos computacionais. Isso é possível através da transferência temporária de processos residentes na memória principal para a memória secundária, liberando espaço para novos processos.

**Tipos de Alocação**

**Alocação Contígua Simples**

A alocação contígua simples foi implementada nos primeiros sistemas operacionais, porém ainda está presente em alguns sistemas monoprogramáveis. Nesse tipo de organização, a memória principal é subdividida em duas áreas: uma para o sistema operacional e outra para o programa do usuário.

**Técnica de Overlay**

Na alocação contígua simples todos os programas estão limitados ao tamanho da área de memória principal disponível para o usuário. Uma solução encontrada para o problema é dividir o programa em módulos, de forma que seja possível a execução independente de cada módulo, utilizando uma mesma área de memória.

**Alocação Particionada**

Sistemas multiprogramáveis já são muito mais eficientes no uso do processador, necessitando, assim, que diversos programas estejam na memória principal ao mesmo tempo e que novas formas de gerência da memória sejam implementadas.

**Alocação Particionada Estática**

Nos primeiros sistemas multiprogramáveis, a memória era dividida em pedaços de tamanho fixo, chamados partições. O tamanho das partições, estabelecido na fase de inicialização do sistema, era definido em função do tamanho dos programas que executariam no ambiente.

**Alocação Particionada Estática Absoluta**

Inicialmente, os programas só podiam ser carregados e executados em apenas uma partição específica, mesmo se outras estivessem disponíveis. Essa limitação se devia aos compiladores e montadores que geravam apenas código absoluto. No código absoluto todas as referências a endereços no programa são posições físicas na memória principal, ou seja, o programa só poderia ser carregado a partir do endereço de memória especificado no seu próprio código.

**Alocação Particionada Estática Relocável**

Com a evolução dos compiladores, montadores, linkers e loaders, o código gerado deixou de ser absoluto e passou a ser relocável. No código relocável, todas as referências a endereços no programa são relativas ao início do código, e não a endereços físicos de memória. Os programas puderam ser executados a partir de qualquer partição.

**Tabela de Alocação de Partições**

Para manter o controle sobre quais partições estão alocadas, a gerência de memória mantém uma tabela com o endereço inicial de cada partição, seu tamanho e se está em uso. Sempre que um programa é carregado para a memória, o sistema percorre a tabela, na tentativa de localizar uma partição livre onde o programa possa ser carregado.

**Alocação Particionada Dinâmica**

Na alocação particionada dinâmica, ou variável, foi eliminado o conceito de partições de tamanho fixo. Nesse esquema, cada programa utilizaria o espaço necessário, tornando essa área sua partição.